About consistent superstrings

Yu-Hsuan Hsieh

Department of Mathematics,
National Tsing Hua University
benny01237@gmail.com

May 28, 2023

1. Outlines
2. CSS problem
   - Origin of CSS problem
   - Formulation of CSS problem
   - Set-up of CSS problem
3. Graph model
   - construction of graph model G_{AC}
   - example of graph model G_{AC}
   - Some notations for graph model G_{AC}
   - lemmas of the graph model
   - Final graph model G_{CSS}
4. Algorithm for finding SCSS
   - Example of the algorithm
5. References

Outlines

- Consistent Superstring Problem(CSS problem)
- Graph model related to Aho-Corasick algorithm
- Lemmas of the graph model
- Algorithm for finding shortest consistent superstrings

Origin of CSS problem

CSS problem can be view as a generalization of shortest superstring problem(simply, SSP).
SSP arose because of the difficulty in sequencing the whole molecule directly.
We reconstruct the DNA molecule based on the fragments of small length cut randomly from the DNA molecule.

Formulation of CSS problem

Let Σ be a set of alphabet of constant size.
Let Σ^* be the set of strings over Σ.
Given two sets $\mathcal{N} = \{x_1, \ldots, x_t\}, \mathcal{P} = \{y_1, \ldots, y_r\}$ of strings, where $\mathcal{N}$ is called negative strings and $\mathcal{P}$ is called positive strings, our goal is to find shortest superstrings α for $\mathcal{N}$ and $\mathcal{P}$.
That is, each y_i is a substring of α and each x_i is not a substring of α.

Set-up of CSS problem

To avoid trivial cases, we require $\mathcal{N}$ and $\mathcal{P}$ satisfy:

1. $\forall x_i, x_j \in \mathcal{N}, i \neq j$, x_i is not a substring of x_j.
2. $\forall y_i, y_j \in \mathcal{P}, i \neq j$, y_i is not a substring of y_j.
3. $\forall x_i \in \mathcal{N}, y_j \in \mathcal{P}$, x_i is not a substring of y_j.

In addition, Jiang and Timkovsky require $\mathcal{N}$ and $\mathcal{P}$ satisfy:

4. $\forall x_i \in \mathcal{N}, y_j \in \mathcal{P}$, y_j is not a substring of x_i.
5. $\forall \sigma \in \Sigma$, there is a negative string ending up by σ.

$\mathcal{N}$ and $\mathcal{P}$ are called inclusion free if they satisfy 1-4.
$\alpha[i]$ denotes the i-th character of the string α.
A prefix of a string α is a substring $\alpha[1] \cdots \alpha[j] = \alpha[1 \cdots j]$, where $1 \leq j \leq |\alpha|$.
A suffix of a string α is a substring $\alpha[j] \cdots \alpha[|\alpha|] = \alpha[j \cdots |\alpha|]$, where $1 \leq j \leq |\alpha|$.

Graph model related to Aho-Corasick algorithm

Here gives the construction of a graph model $G_{AC} = (V, E)$.

1. Vertex set V:
   Consider a set $\mathcal{T}$ consisting of prefixes of all strings in $\mathcal{N} \bigcup \mathcal{P}$. Let λ be the empty string, which is both prefix and suffix of any string. For each string α in $\mathcal{T}$, add one vertex $ver(\alpha)$ in V. In particular, $ver(\lambda) = v_0 \in V$. For each vertex $v \in V$, $str(v)$ denotes the corresponding string in $\mathcal{T}$.

Graph model related to Aho-Corasick algorithm

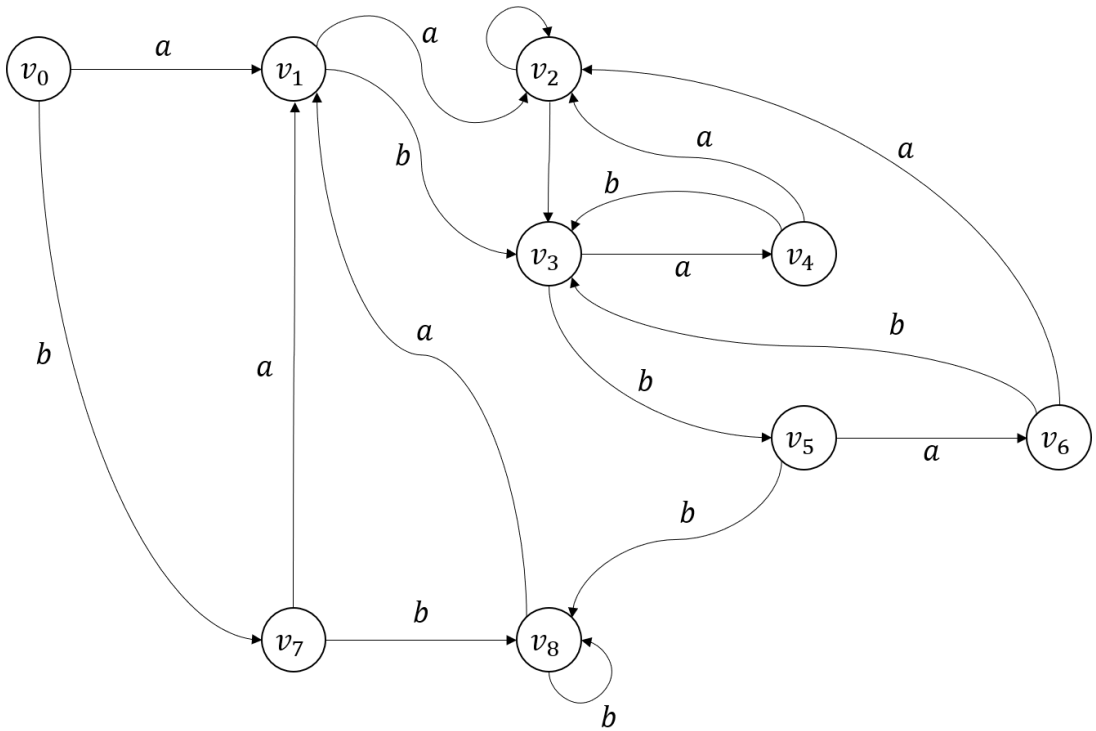
Here gives the construction of a graph model $G_{AC} = (V, E)$.

2 Edge set E:
For $\alpha \in \mathcal{T}, \sigma \in \Sigma$, define $\delta : \mathcal{T} \times \Sigma \longrightarrow \mathcal{T}$ by $\delta(\alpha, \sigma)$ being the longest suffix $\beta \in \mathcal{T}$ of $\alpha\sigma$.
A directed edge from $ver(\alpha)$ to $ver(\beta)$ labeled with σ is defined if and only if $\beta = \delta(\alpha, \sigma)$.

Obviously, the sizes of both vertex set and edge set are $O(n + p)$, where n is the sum of lengths of all strings in $\mathcal{N}$, p is the sum of lengths of all strings in $\mathcal{P}$.

Example of graph model G_{AC}

We give an example for $\mathcal{N} = \{aa, abba\}$ and $\mathcal{P} = \{aba, bb\}$ over $\Sigma = \{a, b\}$. In this case, $\mathcal{T} = \{\lambda, a, aa, ab, aba, abb, abba, b, bb\}$. The vertex set $V = v_0, \ldots, v_8$ represent the strings in $\mathcal{T}$ under the same order.



Some notations for graph model G_{AC}

For $\alpha \in \mathcal{T}$, define $V_s(\alpha) = \{ver(\gamma) : \gamma \in \mathcal{T}, \alpha \text{ is a suffix of } \gamma\}$.
$V_s(\mathcal{P})$ is the union of $V_s(y)$, $\forall y \in \mathcal{P}$.
$V_s(\mathcal{N})$ is the union of $V_s(x)$, $\forall x \in \mathcal{N}$.
For a path $A = (u_1, \cdots, u_k)$ in G_{AC}, $pstr(A) = l_1 \cdots l_{k-1}$, l_i is the label of the edge (u_i, u_{i+1}), is the string corresponding to A.
Similarly, for a string α, there are many paths corresponding to it, denoted by $path(\alpha)$. The one starts from $v_0 = ver(\lambda)$ is called λ-$path(\alpha)$. The existences of $path(\alpha)$ and λ-$path(\alpha)$ are provided by the construction of G_{AC}.
A path is called $\mathcal{P}$-path if it is a $\lambda-$path that passes at least one vertex in $V_s(y), \forall y \in \mathcal{P}$.
A path is called $\mathcal{N}$-path if it is a $\lambda-$path that passes no vertex in $V_s(\mathcal{N})$.

Lemmas of the graph model

Lemma (1)

For a string $\beta \in \mathcal{N} \bigcup \mathcal{P}, \alpha$ includes β as a substring if and only if λ-path(α) passes a vertex $v \in V_s(\beta)$.

Lemma (2)

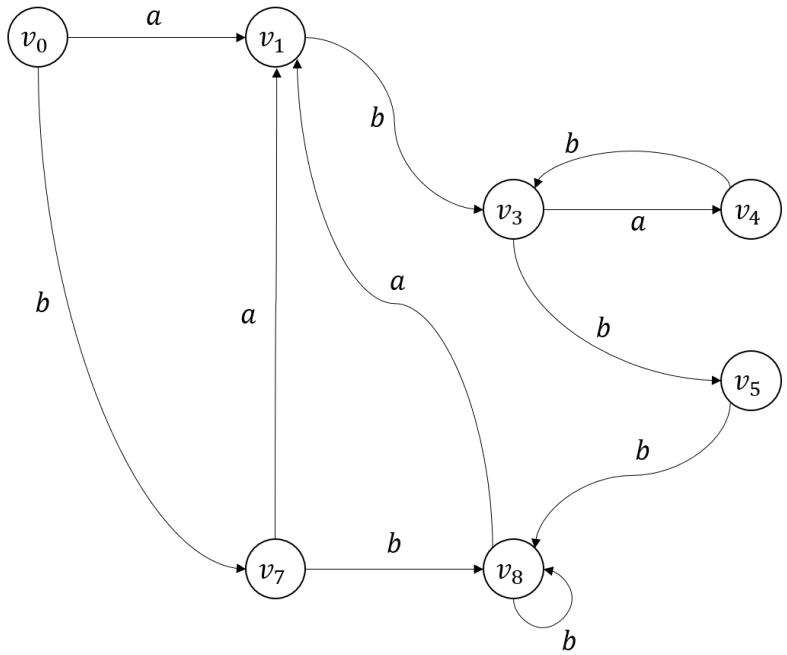
A string α is a common non-superstring of $\mathcal{N}$ if and only if λ-path(α) is an $\mathcal{N}$-path in G_{AC}.

Lemma (3)

A string α is a consistent superstring of $\mathcal{N}$ and $\mathcal{P}$ if and only if λ-path(α) is both an $\mathcal{N}$-path and a $\mathcal{P}$-path in G_{AC}.

Goal of graph model

The graph G_{AC} is just an intermediate graph. Define $G_{CSS} = (V', E')$ such that $V' = V - V_s(\mathcal{N})$ and $E' = E - \{(v_i, v_j) : v_i \in V_s(\mathcal{N}) \text{ or } v_j \in V_s(\mathcal{N})\}$.
That is, G_{CSS} is obtained from G_{AC} by removing all corresponding vertices of negative strings and edges incident with them.



Algorithm for finding SCSS

The algorithm is based on the following corollary:

Corollary

A string α is a consistent superstring of $\mathcal{N}$ and $\mathcal{P}$ if and only if λ-path(α) is a $\mathcal{P}$-path in G_{CSS}.

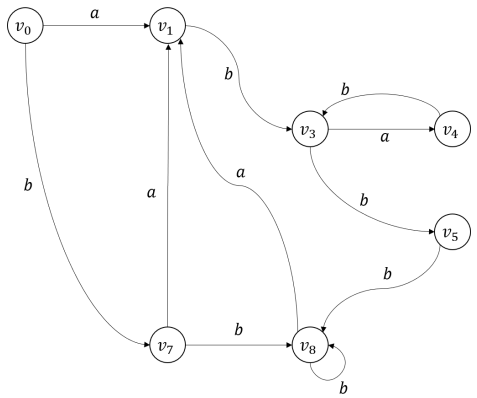
We find $\mathcal{P}$-paths by using the vertices in $V_s(\mathcal{P})$.
Choose w_i from $V_s(y_i)$, for each $1 \leq i \leq r$.
A $\mathcal{P}$-sequence is a permutation π of $\{w_i\}$. If there is a $\mathcal{P}$-path passes $\{w_i\}$ in order in π, then we say the $\mathcal{P}$-path embeds the $\mathcal{P}$-sequence π and π is said to be valid.
For each $\mathcal{P}$-path, there is at least one $\mathcal{P}$-sequence embedded by it.
So, we can find $\mathcal{P}$-paths by considering $\mathcal{P}$-sequences.

Algorithm for finding SCSS

The algorithm consists of three phases:

1. Construct G_{CSS}.
2. Find the shortest $\mathcal{P}$-path using $\mathcal{P}$-sequences.
3. Compute the SCSS using the shortest $\mathcal{P}$-path.

Example of the algorithm



In this example, $V_s(aba) = \{aba\}$, $V_s(bb) = \{abb, bb\}$.
The four $\mathcal{P}$-sequences are :

1. $\pi_1 = (aba, abb) = (v_4, v_5)$
2. $\pi_2 = (aba, bb) = (v_4, v_8)$
3. $\pi_3 = (abb, aba) = (v_5, v_4)$
4. $\pi_4 = (bb, aba) = (v_8, v_4)$

Example of the algorithm

The shortest $\mathcal{P}$-paths embedding each $\mathcal{P}$-sequences are:

1. $A_1 : v_0 - v_1 - v_3 - v_4 - v_3 - v_5$
2. $A_2 : v_0 - v_1 - v_3 - v_4 - v_3 - v_5 - v_8$
3. $A_3 : v_0 - v_1 - v_3 - v_5 - v_8 - v_1 - v_3 - v_4$
4. $A_4 : v_0 - v_7 - v_8 - v_1 - v_3 - v_4$

Obviously, A_1 and A_4 are the shortest. And the string corresponding to them are *ababb* and *bbaba*, respectively, which are both the SCSS of $\mathcal{N} = \{aa, abba\}$ and $\mathcal{P} = \{aba, bb\}$

References

Kim, J.W., Choi, S., Na, J.C., Sim, J.S. (2009). Improved Algorithms for Finding Consistent Superstrings Based on a New Graph Model. In: Dong, Y., Du, DZ., Ibarra, O. (eds) Algorithms and Computation. ISAAC 2009. Lecture Notes in Computer Science, vol 5878. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-10631-6$_1$19